memory bits)
memory increments)

Preliminary Specification

PDP-11 Instruction Set

January 24, 1969

CONFIDENTIAL

DIGITAL EQUIPMENT CORPORATION

PM 21-19

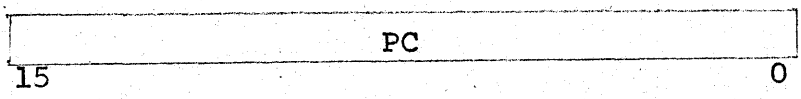
I. Introduction

The PDP-11 is a 16-bit small computer. It operates on bytes (8 bits) or words (16 bits). The memory is byte addressable, to a maximum of 65536 bytes. Instructions are one, two, or three bytes (8, 16, 24 bits). This memo describes the processor registers, the memory reference instruction addressing and the five groups of machine instructions - memory reference, operate, conditional jump, add to register and external.

II. Processor Registers

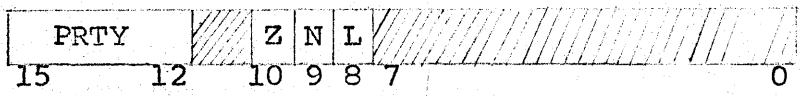
The PDP-11 has 8 16-bit hardware registers.

A. Program Counter



Prior to each instruction, the program counter contains the address of the instruction. During the execution it contains the address of the next instruction. The program counter itself is addressable at (177776)8.

B. Status Register



Bits 15 through 12 of the status register contain the current processor priority. Bits 10 through 8 contain the condition codes. Bit 11 is unused and will always be zero if examined. Bits 7-0 may not be referenced.

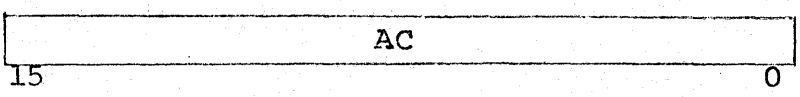
1. Condition Codes

Bits 10 through 8 of the status register are called the condition codes. They are set according to the results of most instructions. L (bit 8) is complemented as a result of carry out from the adder. N (bit 9) is set (reset) if the result of an arithmetic instruction is negative (non-negative). Z (bit 10) is set (reset) if the result of an arithmetic operation is zero (non-zero).

2. Priority

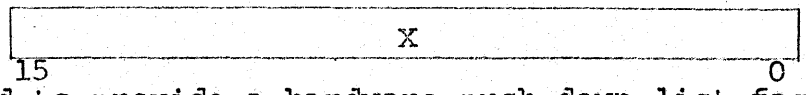
Bits 15 through 12 of the status register determine the priority of the currently running program as a binary number from 0 to (17)8.

C. Accumulator



This register is used both as an operand and as the result register for most arithmetic instructions. In addition to being implicitly addressed by these instructions, it also may be explicitly addressed at location (177772)8.

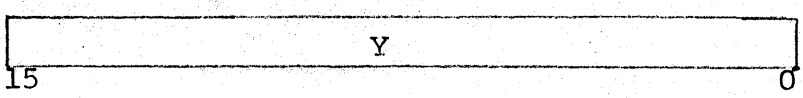
D. X-Register



The X-Register is used to provide a hardware push down list facility.

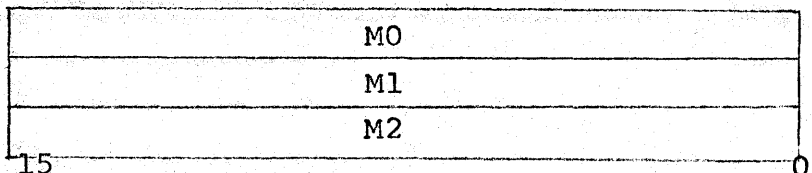
It may also be employed as a general index register. It is incremented by two during the subroutine call instruction and decremented by two during the subroutine return. It is also incremented and decremented by the ATX instruction to provide reentrant subroutines. It may be explicitly addressed at (177766)8.

E. Y-Register



This register is used for subroutine argument communication and as a general index register. For subroutine communication, it is set to the return address by the subroutine calling instruction. The Y-register may also be explicitly addressed at (177770)8.

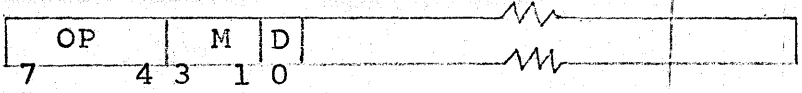
F. "Mini" Registers



The "mini" registers are three hardware 16-bit registers used by the program as temporary storage. They are normally addressed in the "mini" mode described below in Section III. In addition, MØ, M1, M2 may be explicitly addressed at (177760)8, (177762)8, (177764)8, respectively.

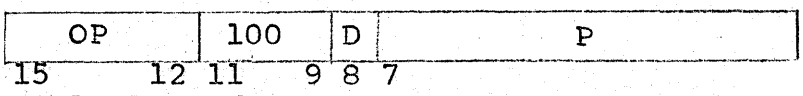
why not in page0?

III. Memory Reference Instruction Addressing



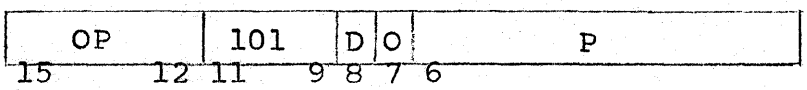
Each memory reference instruction consists of one, two or three bytes. The first byte is called the instruction byte. Successive bytes, if any, contain either address information or data. The operation code (OP) is in the range of Ø through (14)8. The mode bits (M) determine the address computation. The deferred bit (D) specifies whether the address is to be deferred one level indirectly.

A. Relative Addressing



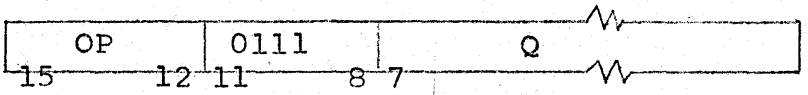
The 2's complement offset P is added to the program counter to form an effective address which is in the range (-200, +177) relative to the program counter.

B. Page Ø Addressing



P is used to form an address in the range (0, 177).

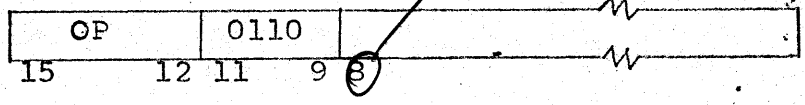
C. Full Addressing



Q is taken as a full 16-bit address. This address mode cannot be deferred.

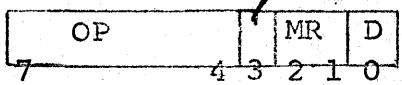
? only 8bits

D. Immediate Addressing



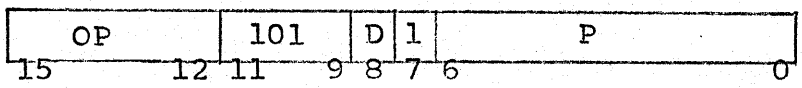
The operand is taken directly from the byte or word following the instruction byte. The operand is a byte or a word depending on the operation code (see Section IV). This addressing mode may not be deferred.

E. "Mini" Addressing



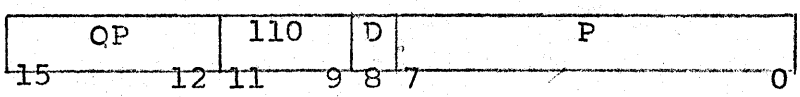
The effective address before deferral is MØ, M1, M2 depending on whether MR is Ø, 1, 2 respectively. If D is set, an "autoincrement" is added to the mini register to determine the effective address. This new effective address also replaces the contents of the mini register. The autoincrement is Ø, 1 or 2 depending on the instruction (see Section IV).

F. External Addressing



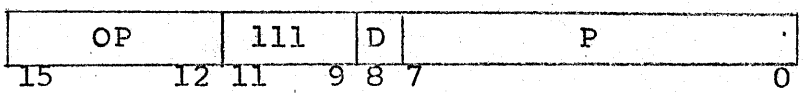
In this mode, the high order 9 bits of the effective address before deferral are all ones. The low order 7 bits are P. Thus the address range before deferral is (177600-177777). Input/output and external devices are typically assigned in this area - thus the external addressing mode allows input/output or external references in a two byte instruction.

G. X-Indexed Addressing



The effective address before deferral is the sum of the 2's complement number P and the contents of the X-register. This provides a range of (-200, +177) relative to the contents of X.

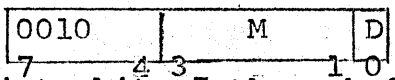
H. Y-Indexed Addressing



The effective address, before deferral, is the sum of the 2's complement number P and the contents of the Y-register. This provides a range of (-200, +177) relative to the contents of Y.

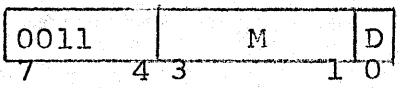
IV. Memory Reference Instruction Group

A. Load Byte (LDB)



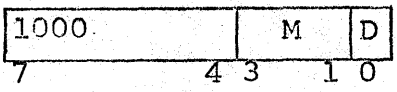
The effective byte is loaded into bits 7 through 0 of the accumulator. The high order bit of the effective byte is extended through bits 15-8 of the accumulator. L is not effected. N is set to bit 15 of the effective byte. Z is set to one (zero) if the effective byte is zero (non-zero). The autoincrement is one.

B. Load Word (LDW)

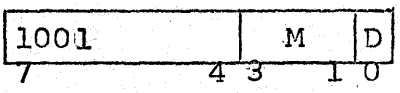


The effective word is loaded into the accumulator. L is not changed.

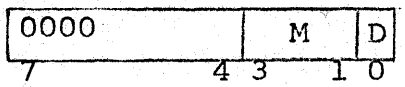
N is set to bit 15 of the effective word. Z is set to one (zero) if the effective word is zero (non-zero). The autoincrement is two.

C. Store Byte (STB) 

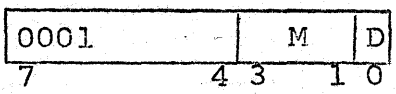
The contents of accumulator bits 7 through 0 are stored in the effective byte. The condition codes are not affected. The autoincrement is one.

D. Store Word (STW) 

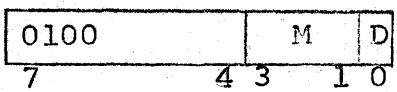
The contents of the accumulator are stored in the effective word. The condition codes are not affected. The autoincrement is two.

E. Add Byte (ADB) 

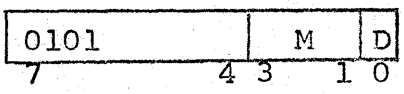
The effective byte is treated as a 2's complement 8-bit quantity and added, with sign extended, to the contents of the accumulator. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero). The autoincrement is one.

F. Add Word (ADW) 

The effective word is added to the contents of the accumulator. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero). The autoincrement is two.

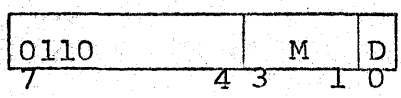
G. Compare Byte (CPB) 

The 2's complement of the accumulator and the effective byte are added together. The effective byte is treated as a 2's complement 8-bit quantity. The result is not stored in any register, but is used to set the condition codes. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero). The autoincrement is one.

H. Compare Word (CPW) 

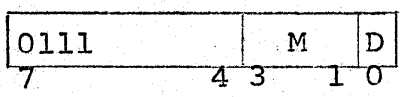
The 2's complement of the accumulator and the effective word are added together. The result is not stored in any register, but is used to set the condition codes. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero). The autoincrement is two.

I. And Byte (ANB)



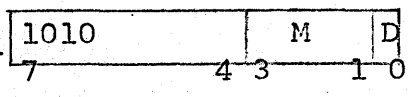
The effective byte is anded to the contents of the accumulator bits 7 through 0. Bits 15 through 8 of the accumulator are not affected. L is not changed. N is set to the sign of the accumulator. Z is set to one (zero) if the accumulator is zero (non-zero) after the operation. The autoincrement is one.

J. And Word (ANW)



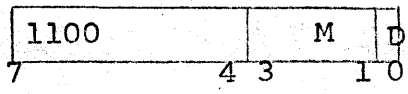
The effective word is anded to the contents of the accumulator. L is not changed. N is set to one (zero) if the result is negative (non-negative). Z is set to one (zero) if the result is zero (non-zero). The autoincrement is two.

K. Increment (INC)



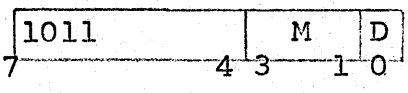
The effective word is incremented by one. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero). The autoincrement is two.

L. Jump (JMP)



The effective address replaces the contents of the program counter. The condition codes are not affected. The autoincrement is zero.

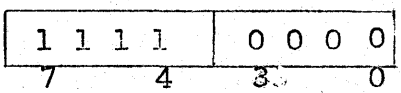
M. Jump to Subroutine (JSR)



The program counter is stored in the word which has the address specified by the contents of the X-register. The X register is then incremented by two. The old program counter is also stored in the Y-register. The effective address then replaces the contents of the program counter. The condition codes are not affected. The auto-increment is zero.

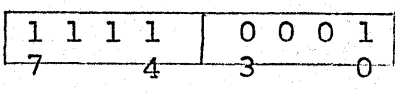
V. Operate Instruction Group

A. No Operation (NOP)



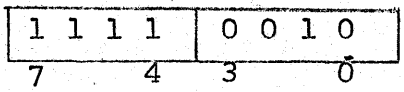
The program counter is advanced one byte. The condition codes are not changed.

B. Increment Accumulator (IAC)



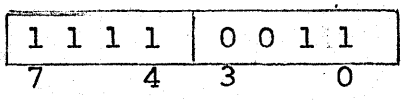
The accumulator is incremented by one. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

C. Complement Accumulator (CMA)



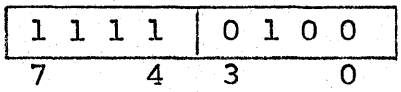
The contents of the accumulator are replaced by the ones complement of the original contents. The L bit is not affected. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

D. Negate (NEG)



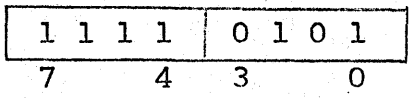
The two's complement of the accumulator replaces the original accumulator contents. L is complemented if the original contents of the accumulator are zero. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

E. Clear Accumulator (CLA)



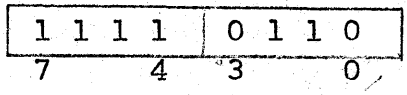
The contents of the accumulator are set to zero. L is not affected. N is set to zero and Z is set to one.

F. Plus One to Accumulator (POA)



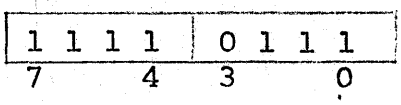
The accumulator is set to plus one. L is not effected. Both N and Z are set to zero.

G. Minus One to Accumulator (MOA)



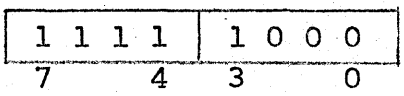
The accumulator is set to minus one. L is not affected. N is set to one and Z is set to zero.

H. Clear Accumulator and Complement Link (CLC)



The accumulator is cleared and the L bit is complemented. N and Z are set to zero.

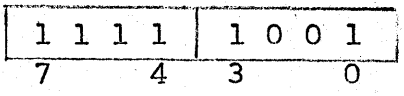
I. Rotate Accumulator Right (RAR)



The L bit and the accumulator are treated as a single 17 bit circular register. The conbination is rotated right one bit position (L goes into accumulator bit 15 and accumulator bit Ø goes into L). N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

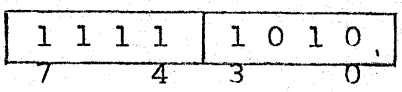
can't do better?

J. Rotate Accumulator Left (RAL)



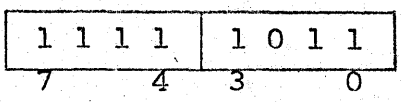
The L bit and the accumulator are treated as a single 17 bit register. The combination is rotated left one bit position (L goes to accumulator bit Ø and accumulator bit 15 goes to L). N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

K. Clear Link (CLL)



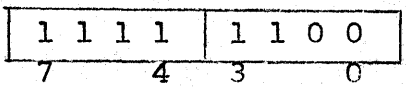
L is set to zero. N and Z are not affected.

L. Complement Link (CML)



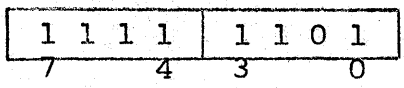
The L bit is complemented. N and Z are not affected.

M. Return (RTN)



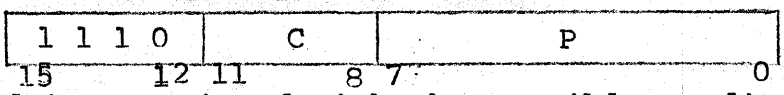
The word at the address which is two less than the contents of X replaces the program counter. The contents of the X-register are then decremented by two. This instruction is the return for JSR.

N. Return from Interrupt (RTI)



The program counter is replaced by the word in the address three less than the contents of X. The status register bits 15-8 are replaced by the byte at the address one less than the contents of X. The X-register is then decremented by three. This instruction is the return for EXC and also is the last instruction in each interrupt handling routine.

VI. Conditional Jump Instruction Group



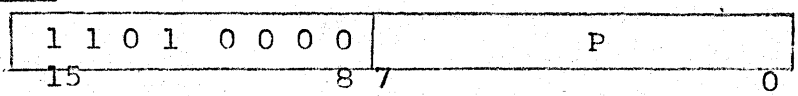
The C field is associated with the possible conditions of the L, N and Z codes. If the specified condition is met, the offset P is added as a two's complement number to the program counter. This provides a range of (-200, +177) relative to the current program counter. If the condition is not met, the next instruction is executed. The condition codes themselves are not affected.

The mnemonics, C field and the conditions tested are:

C	mnemonic	meaning	condition
04	JEQ	jump on equal	Z
14	JNE	jump on not equal	not Z
02	JLT	jump on less than	N
16	JGT	jump on greater than	not N and not Z
06	JLE	jump on less than or equal	N or Z
12	JGE	jump on greater or equal	not N
01	JLS	jump on link set	L
11	JLR	jump on link reset	not L

VII. Add to Register Group

A. Add to MØ (AMØ)

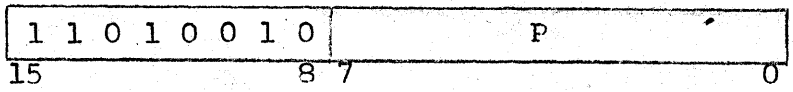


P is treated as an 8-bit two's complement number and added to the

Signed?

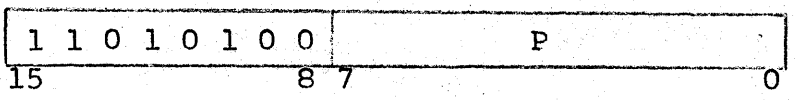
contents of MØ. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

B. Add to M1 (AM1)



P is treated as an 8-bit two's complement number and added to M1. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

C. Add to X-Register (ATX)



P is treated as an 8-bit two's complement number and added to X. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero (non-zero).

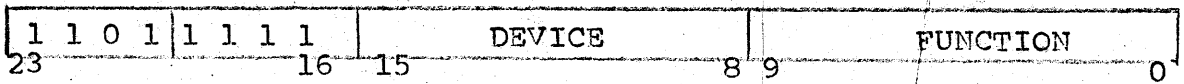
D. Add to Y-Register (ATY)

P is treated as an 8-bit two's complement number and added to Y. L is complemented if the operation causes a carry out of bit zero of the adder. N is set to the high order bit of the result. Z is set to one (zero) if the result is zero(non-zero).

VIII. External Instruction Group

This group of instructions allows the program to control certain devices in the system. The processor itself is device zero - a reference to the processor in an external instruction causes a programmed interrupt. The console is device (377)8 - an external instruction to the console causes a halt. Other device numbers may be assigned to extended arithmetic elements, input/output devices or user equipment. The function of the external instructions in these cases depends on the device.

A. External Control (EXC)



1. Interrupt of Processor

This instruction causes the processor to "interrupt itself" - that is, respond as if the interrupt had occurred from an external device. It occurs if the contents of DEVICE are zero. The program counter is stored in the location specified by the contents of the X-register. The status byte is stored in the location which is two greater than the contents of the X-register. The X-register itself is incremented by three.

The processor resumes operation as determined by a three byte "status block" which begins at the address with bits 15-8 zero and bits 7-0 FUNCTION. The first two bytes of the status block replace the program counter and the third byte becomes the new status byte.

2. Halt

If DEVICE is (377)8, the system stops. Condition codes are not affected.

3. Input/Output Control

If DEVICE refers to an input/output device the operation of the instruction is determined by the device.

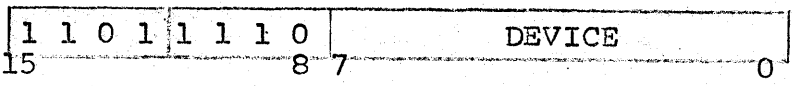
4. Extended Arithmetic References

If the referenced device is an extended arithmetic unit, that unit performs the appropriate operation and returns control. The instruction format is determined by the referenced unit.

5. Trap

If DEVICE specifies a non-existant unit, and if the "time-out" option is installed, the effect is the same as an external control instruction with DEVICE and FUNCTION zero. This is used to simulate unimplemented extended operation codes by software or firmware programs. The contents of the word at location zero must be the starting address of a software trap handler.

B. External Select (EXS)



The EXS instruction functions the same as an EXC (external control) with a FUNCTION of zero.

protect system